XP-001026379

P.D. 01-11-1998 (5)
P. 910-914

# Structured and Content-based Video Browsing

HongJiang Zhang and Wei-Ying Ma
Hewlett-Packard Laboratories
1501 Page Mill Road, Palo Alto, CA 94304-1126

## Abstract

*In this paper, we argue the importance of content-based video browsing and present a set of power tools for video browsing. We propose a content representation scheme that captures both structure and visual content of video to support content-based video browsing. To extracts video structure and content primitives and visual abstraction needed by the representation scheme, a set of robust algorithms for video parsing, abstraction and clustering have been developed and is presented in this paper.*

## 1 Introduction

In the last few years, substantial research efforts have been devoted to visual data content analysis and retrieval. Lately, MPEG committee started a new effort to standardize multimedia content descriptors in response to the needs in video applications, especially interactive video and digital libraries, where efficient video indexing and retrieval tools are essential. Many research efforts have been devoted to automatic content-based indexing, mostly using low-level features and structural information. However, due to the limitation the low-level visual features in representing semantic content of video, the performance of most proposed content-based retrieval algorithms are far from satisfactory [1].

On the other hand, interactive video browsing tools that utilize the structural and content information of video can be very effective for quick relevance assessment of video material. How can we spend only few minutes to view an hour of video and still have a fairly correct perception of what its contents are like? Browsing is a means that is more suitable to address this need. By browsing, we mean an informal but quick access of content that often lacks any specific query. Browsing is also essential for retrieval of video. It serves as an aid to *formulating queries*, making it easier for the user to "just ask around" in the process of figuring out the most appropriate query to pose.

Recent works in video parsing provide a foundation for building interactive and content-based video browsing tools. In the parsing process, temporal structure in terms of shots and scenes are extracted, which provides the fundamental information for structured video browsing. Browsing can be made even more efficient if shots and scenes are represented visually in some abstracted forms, such as key-frames and visual highlights.

In this paper, we present a video content representation scheme to support structured and content-based video browsing, and a set of video structure parsing and content abstraction algorithms to extract primitives of the representation scheme. A set of video browsing tools that utilizes the information represented in the proposed scheme is also discussed. Finally, we will present some thoughts on possibilities to support structured and content-based video browsing in the new MPEG-7 efforts.

## 2 Content representation, parsing and abstraction

Content-based video browsing tools should support different nonlinear approaches to accessing video source data, either sequential or random access. Also, these tools should accommodate different levels of granularity. Sequential browsing is often needed in quickly viewing, for instance, an hour of video in a few minutes and still have a fairly correct perception of what its contents are like. Visual highlights (also referred as summaries or previews) are designed to support such browsing. Content-based random access requires information about video structure. Therefore, it is essential to build a content representation and compute the representation primitives, based on which the content of video data can be classified, indexed and compared.

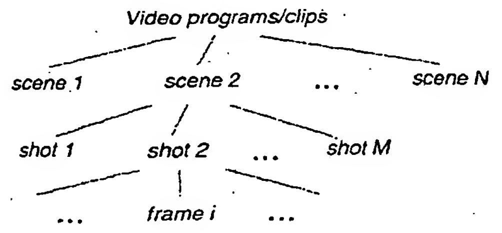

**Figure 1:** The hierarchical structure of video content.

### 2.1 A hierarchical content representation

As illustrated in Figure 1, a video sequence or document can be decomposed into three structural levels: programs, scenes and shots. A program is composed of a

BNSDOCID: <XP_________1026379A__I_>

set of scenes. Scenes are further partitioned into shots. Each shot contains a sequence of frames recorded contiguously and representing a continuous action in time or space. The representation of a video document should have the same structural levels. Therefore, we propose the following structured representation scheme.

*Sequence*

*Sequence-ID*: $x$ (a unique index key)
*Scenes*: {Scene(1), Scene (2), ... Scene (L)}
***Highlight: A***

*Scene*

*Scene-ID*: $y_l$ (a unique index key of the scene $l$)
*Shots*: {Shot(1), Shot(2), ... Shot(M)}
***Key-frames*: {KF$_l$(1), KF$_l$(2), ..., KF$_l$(I)}**

*Shot*

*Shot-ID*: $z_m$ (a unique index key of the shot $m$)
*Primitives*: $\{f_s(1), f_s(2), \dots, f_s(N)\}$
***Key-frames*: {KF$_m$(1), KF$_m$ (2), ..., KF$_m$ (J)}**

This representation scheme maintains the hierarchical structure of video: sequences, scenes and shots, each is defined by a set of primitives. Therefore, this structure meets the need for random and nonlinear browsing: one can easily move from one level of detail to the next and skip from one segment of video to another. Note that the bolded items are primitives based on visual abstractions of a sequence, which are not usually contained explicitly in conventional video representation schemes. That is, an abstracted representation is associated with each level of the structure hierarchy: highlights associated at the top level; two levels of key-frames are associated with the scene and shot level, respectively. These primitives are included to make the representation especially useful in content-based video browsing.

Key-frames are still images extracted from original video data that best represent the content of shots in an abstract manner. Key-frames have been frequently used to supplement the text of a video log, though they were selected manually in the past [2]. Key-frames, if extracted properly, are a very effective visual abstract of video contents and are very useful for fast video browsing. A video summary, such as a movie preview, is a set of selected segments from a long video program that highlight the video content, and it is best suited for sequential browsing of long video programs. Just like keywords and abstracts in text document browsing and indexing, these two forms of video abstraction represent the landscape or structure of video sequences and entries to shots, scenes or stories that make fast and content-based video browsing possible. Apart from browsing, key-frames can also be used in representing video in retrieval: video index may be constructed based on visual features of key-frames, and queries may be directed at key-frames using query by image retrieval algorithms.

In our representation scheme, the abstract is also hierarchical. That is, the key-frames at the scene level could be a sub-set of the key-frames of all shots comprising the scene; thus, $KF_l(i)$ could be a set of *pointers* to $KF_m(j)$. That is, $KF_l(j)$ is a more abstracted set extracted from $KF_m(j)$. Similarly, the video highlights, $A$, as part of the sequence representation, could be a set of pointers to the highlight segments extracted by using video summarization approaches.

Give the representation structure, another issue is to use which set of low-level visual features as content primitives, so that similar shot can be clustered into groups, thus, support class-based browsing. In our representation, the visual features are divided into two groups: those derived from key-frames only, i.e. *key-frames-based*, and those, *shot-based*, derived from all frames of a shot, such as the statistical motion measures, and other temporal features as proposed by Zhang, *et al.* [3]. The key-frame-based features capture representative information at a few sampling points where key-frames are located, while shot-based features provide information about activities, and motion complexity that is useful in measuring temporal content similarity. These two groups of features constitute a complete set of content primitive for video shots and form the basis for shot based video clustering and indexing.

Therefore, we can add one more extension for each of key-frames into the representation scheme as in below:

*Key-frames:*
*KF-ID:* **KF$_m$(j)**
*Primitives*: $\{f_{KF}(1), f_{KF}(2), \dots, f_{KF}(K)\}$.

Although here we focus on video content representations based upon low-level features, this scheme can also accommodate high-level semantic feature. For instance, one can include high-level semantic features, such as objects, e.g. news-anchor, in a key-frame. Focused on home video, we have developed a video parsing system that detects and tracks human faces in video and using faces as a semantic feature for video browsing and retrieval [4].

## 2.2 Video structure parsing and abstraction

Extracting structure primitives in the proposed representation scheme is the task of video segmentation, that involves the detecting of temporal boundaries between scenes and between shots. A robust video segmentation algorithm should be able to detect all kinds of shot and scene boundaries with good accuracy. There have been many shot segmentation algorithms proposed, but very few for scene segmentation. We have developed a robust shot segmentation algorithm that operates on MPEG compressed video directly with very high accuracy and faster than real-time processing speed [4]. Also, to overcome the bottleneck of threshold selection, a statistic

BNSDOCID <XP________1026379A__I_>

framework has been developed with which no threshold is needed in shot segmentation [5].

There are also a few reported algorithms for automatic extraction of key-frames. These algorithms can be categorized into three coarse categories: cumulative difference comparison [3], clustering-based [6] and the optimal assignment [7]. However, a bottleneck in these algorithms is to determine the optimal number of key-frames for a given video sequence. To address this issue, we have developed an approach to video abstraction using an unsupervised clustering framework. The basic idea here is to recognize the natural cluster structure of the visual content, if any, in a video sequence. This is achieved by applying a cluster validity analysis [8].

Similar to the clustering-based key-frame extraction algorithms, in our approach, we apply unsupervised clustering, such as the standard k-means, to all frames of a sequence, with each represented by a feature vector. However, instead of determining the number of clusters in the sequence using a threshold value, we apply the following two-step procedure. First, we classify all frames of a sequence into 1 to $N$ sets of clusters, where $N$ is the maximum allowed number of partitions within a sequence. Consequently, the cluster validity analysis is applied to determine which of the obtained $N$ different clustering options is the optimal one for the given sequence. That is, for each clustering option, $n$ $(1 \le n \le N)$, a cluster separation measure $\overline{R}(n)$ is computed according to [8] as

$$\overline{R}(n) = \frac{1}{n}\sum_{i=1}^{n} \max_{1\le j\le n \,\wedge\, i\ne j}\left(\frac{s_i + s_j}{M_{ij}}\right) \tag{2}$$

where

$$s_i = \left\{\frac{1}{T_i}\sum_{v=1}^{T_i} |F(v) - F(c_i)|^q\right\}^{1/q} \tag{3.1}$$

$$M_{ij} = \left\{\sum_{k=1}^{D} |f_k(c_i) - f_k(c_j)|^p\right\}^{1/p} \tag{3.2}$$

The value $s_i$ is called *dispersion* of the cluster $i$, $M_{ij}$ the *Minkowski metric* of the centroids characterizing the clusters $i$ and $j$. $T_i$ is the number of elements in a cluster.

$\overline{R}(n)$ is a system-wide average of the similarity measures of each cluster with its most similar cluster. Consequently, the optimal cluster structure, given all the clustering options with $(2 \le n \le N)$, would be the one, for which the $\overline{R}(n)$ curve has its minimum.

$$n_{opt} \Leftarrow \min_{1\le n\le N}\left(\overline{R}(n)\right) \tag{4}$$

However, note that the $\overline{R}(n)$ values are computed only for values $2 \le n \le N$ due to the fact that the denominator in (2) must be nonzero. Consequently, the decision to assign only one cluster to a sequence can be made only by eliminating all other possible partitioning. Naturally, if $n_{opt}$=2, there is the need to examine if the given sequence is a highly stationary one whose frames falls into one natural cluster, i.e. $n_{opt}$=1. Therefore, when $n_{opt}$ =2 determined by (4), we further compare the average intra-cluster variance of these two cluster with that of clustering all frames into a single cluster. If the difference is insignificant, then we will choose $n_{opt}$=1, i.e. choosing only one key-frame for the given sequence.

Once the optimal number of clusters is available, the frames closest to each cluster centroid can be taken as the most representative one for the given sequence. We have also extended the proposed approach to key-frame extraction to extraction of video highlights [9].

## 2.3 Shot clustering for browsing

Clustering video segments, such as shots, into groups or classes, each of which contains similar content, is essential to building an index of shots for content-based retrieval and browsing. The first step in clustering is to define content similarity between video sequence. Video content similarity can be defined based on key-frame-based features, shot-based temporal and motion features, object-based features, or a combination of the three. With key-frame-based similarity, if two shots are denoted as $S_i$ and $S_j$, their key-frame sets as $K_i = \{ f_{i,m}\ ,\ m= 1, \dots, M\}$ and $K_j = \{f_{j,n},\ n = 1, \dots , N\}$, then the similarity between the two shots can be defined as

$$\mathbf{s}_k(S_i,S_j) = \max[s_k(f_{i,1},f_{j,1}),\dots,s_k(f_{i,1},f_{j,N}),\dots, s_k(f_{i,m},f_{j,1}),s_k(f_{i,m},f_{j,2}),\dots,s_k(f_{i,m},f_{j,N})] \tag{5}$$

where $s_k$ is a similarity metric between two images defined by any one or a combination of the image features. There are totally $MxN$ similarity values, from which the maximum is selected. This definition assumes that the similarity between two shots can be determined by the pair of key-frames which are most similar, and it will guarantee that if there is a pair of similar key-frames in two shots, then the shots are considered similar. Another measure of the similarity between two shots is the average of the most similar pairs of key-frames. The key-frame-based similarity measure can be further combined with the shot-based similarity measure to make the comparison more meaningful for video.

For the purpose of clustering a large number of shots to build an index with different levels of abstractions, partitioning-clustering methods are most suitable. This is because they are capable of finding optimal clustering at each level and more suitable to obtain a good abstraction of data items. Such an approach has been proposed for video shot grouping [10]. This approach is flexible in that different feature sets, similarity metrics and iterative clustering algorithms can be applied at different levels.

## 3 Browsing tools

There are mainly three kinds of browsing tools built based on different structural and content features of video:

1. time-line or story board based browsers
2. light table of video icons;
3. hierarchical browser;

Time-line based browsers have been favored by users of video production and editing systems, for which time-line interfaces have been used for decades. An example of this kind was the one systematized in the strata model proposed by Aguierre-Smith and Davenpoprt [4]. The light-table video browser is often called clip window, in which a video sequence is spread out in space and represented by video icons. This functions like a light table for slides. A window may contain a sequential list of shots or scenes from a video program, a sequence of shots from a scene, or a group of similar or related shots from a video archival. A user can get a sense of the content of a shot from the icon, particularly when salient stills are used. A clip window browser can also be constructed hierarchically similar to the Windows file management systems used in PC operating systems nowadays.

There are also many ways to construct hierarchical video browser. To improve the content accessibility of hierarchical video browsers, we utilize the structural information of video obtained in video parsing as presented in Section 2. One example of video structure-based browsing is shown in Figure 2, where a video program is segmented into its structural components and is accessed as a tree. At the top of the hierarchy, an entire sequence or program is represented by five key-frames, each corresponding to a group (a scene if we can extract it) of consecutive shots. Any one of these segments may then be subdivided to create the next level of the hierarchy (shots in the case shown in Figure 2). As we descend through the hierarchy, our attention focuses on smaller groups of shots, single shots, and finally the key-frames of a specific shot and all frames of a shot. We can also view sequentially any particular segment of video selected from this browser at any level of the hierarchy by launching a video player.

The hierarchical browser shown in Figure 2 is a power tool for fast assessment of the content of a video sequence or programs, since it utilizes the structure information of video. However, the shots at the high levels of this browser are grouped only according to their sequential relations, not their content similarities. Thus, it is not very convenient when to use for browsing a large collection of video clips, where it is helpful if similar shots can be grouped together. To achieve such grouping, we further extend this browser to use similarity information between shots or sequences, probably pre-clustered. That is, when a list of video clips are accessed for browsing, the system clusters shots into classes, each consisting of shots of similar visual content based on either key-frame and/or shot features. After such clustering, each class of shots is represented by a key-frame determined by the centroid of the class, which is then displayed at the higher levels of the hierarchical browser. Figure 3 shows the data structure and browser layout for such a hierarchical browser. With this type of browser, the viewer can get a rough sense of the content of the shots in a class without moving down to lower levels of the hierarchy.

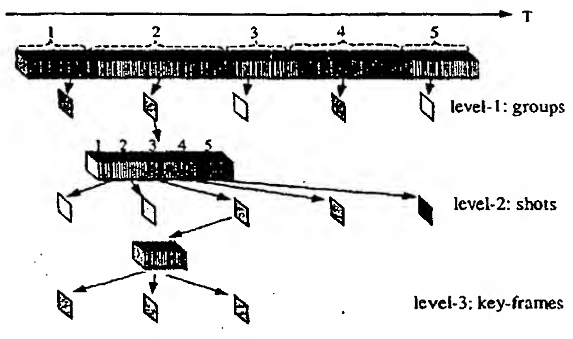

**Figure 2:** Data structure and browser layout for key-frame-based hierarchical browser.

Note that the browsing tools presented make use of key-frames as visual icons to present video content and context. That is, we are able to browse the video content down to the key-frame level without necessarily storing the entire video. This is an important advantage of the key-frame-based browsing schemes, particularly if our storage space and/or bandwidth are limited. Therefore, through the hierarchical browsers, one may browse a large collection of videos before downloading a few of interesting ones identified.

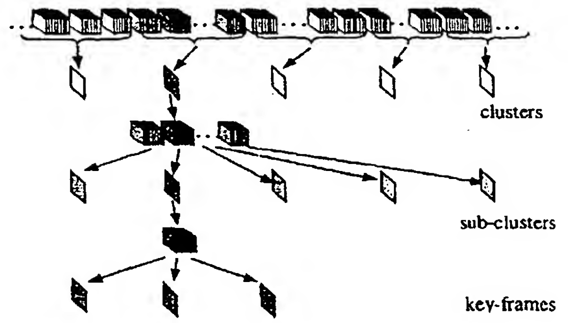

**Figure 3:** Data structure and browser layout for similarity-based hierarchical browser.

Ideally, if we can obtain semantic information of video shots, we can cluster all shot into a tree of semantic concepts instead of the one based on low-level features

BNSDOCID: <XP________1026379A__I_>

shown in Figure 3. Unfortunately, this is not feasible in a general context. However, we could achieve limited success in constrained domain. For organizing home video, we have also developed a video browsing system that integrates human face detection and recognition [4]. This system allows users to browse video sequences according to who is in a particular segment by selecting the key-frames of a particular face. Figure 4 shows a screen dump of this system.

## 4 Conclusion remarks

In this paper, we have addressed the importance of content-based video browsing in video applications and present a set of power tools for video browsing. To support these tools, we have proposed a representation scheme that captures both structure and visual content of video. Also, a set of robust algorithms for video parsing, abstraction and clustering that extract video structure, content primitives and key-frames, have been presented.

An effective content representation scheme is the key to support content-based video retrieval, which is an important task that MPEG-7 standard activities could perform. Unlike its ancestors, MPEG-7, formally departed from narrow defined compressions, will be a standard that addresses "multimedia content description interface," to meet the needs of multimedia content access. To establish a useful and widely acceptable standard for accessing video data, supporting content-based browsing should be essential feature of MPEG-7. To achieve this, a video content representation scheme that at least contains the entries and primitives of the one presented in this paper should be adapted.

## Acknowledgment

We thank Alan Hanjalic for his work on developing the key-frame extraction algorithm presented in this paper.

## References

1. P. Aigrain, H.J. Zhang and D. Petkovic, "Content-based Representation and Retrieval of Visual Media: A State-of-the-Art Review", *Multimedia Tools and Applications*, Kluwer Academic Publishers, Vol.3, No.3, 1996.
2. O'Connor B. C., "Selecting Key-frames of Moving Image Documents: A Digital Environment for Analysis and Navigation", *Microcomputers for Information Management*, 8(2), pp. 119-133, 1991.
3. H. J. Zhang, et al, "Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution", Proc. of ACM Multimedia'95, San Francisco, Nov.7-9, 95, pp.15-24.
4. W.-Y. Ma and H.J. Zhang, "A video segmentation system using MPEG video data", Tech Report, HP Labs, Nov. 98.
5. A. Hanjalic and H.J. Zhang, "Shot Detection based on Statistical Optimization." Tech Report, HP Labs, Sep. 98.
6. Y. Zhuang, *et al*, "Key-frame Extraction Using Unsupervised Clustering", Proc. ICIP'98.
7. A. Hanjalic, R.L Lagendijk. and J. Biemond, "A New Method for Key-frame based Video Content Representation", Image Databases and Multimedia Search, World Scientific, Singapore, 1997.
8. D.L. Davies and D.W. Bouldin, "A Cluster Separation Measure", IEEE Trans. on PAMI, Vol.1, No.2, pp. 224-227, April 1979.
9. A. Hanjalic and H.J. Zhang, "An Integrated Scheme for Automated Video Abstraction using Unsupervised Clustering." Technical Report, HP Labs, September 1998.
10. D. Zhong, H.J. Zhang and S.-F. Chang, "Clustering Methods for Video Browsing and Annotation", *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video Databases IV*, San Jose, Feb. 1996.
11. T. G. Aguierre-Smith and G. Davenport, "The Stratification system: A Design Environment for Random Access Video", *Proc. 3rd Int. Workshop on Network and Operating System Support for Digital Audio and Video*, Nov. 92, pp.250-261.

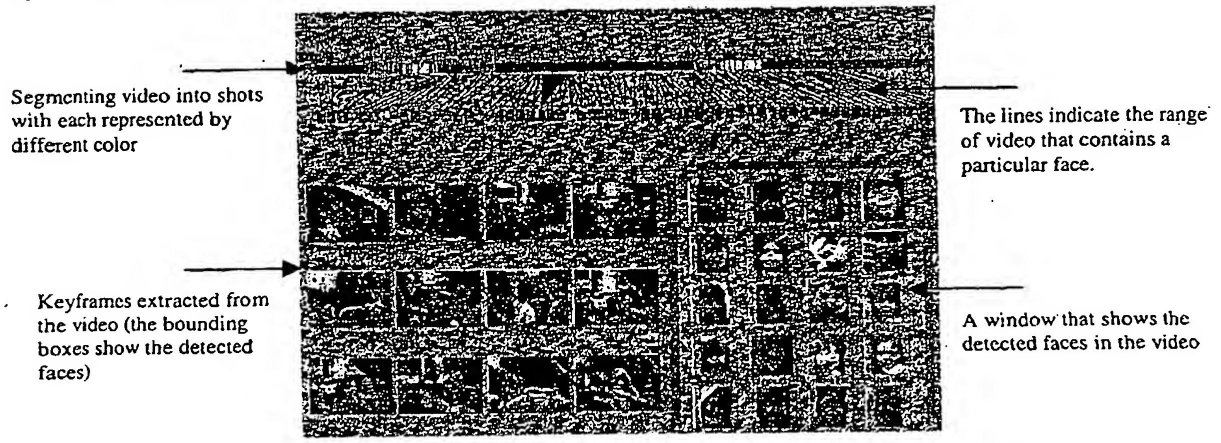
Segmenting video into shots with each represented by different color
Keyframes extracted from the video (the bounding boxes show the detected faces)
The lines indicate the range of video that contains a particular face.
A window that shows the detected faces in the video

**Figure 4.** A video browsing system that integrates human face detection, tracking, and recognition.

BNSDOCID: <XP_______1026379A__I_>

THIS PAGE BLANK (USPTO)